(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 806 756 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
12.11.1997 Bulletin 1997/46

(51) Int Cl.[6]: G09G 5/14

(21) Application number: 97302980.4

(22) Date of filing: 01.05.1997

(84) Designated Contracting States:
FR GB

(30) Priority: 09.05.1996 US 647131

(71) Applicant: INTERNATIONAL BUSINESS MACHINES CORPORATION
Armonk, NY 10504 (US)

(72) Inventor: Bennett, Paul William
Round Rock, Texas 78664 (US)

(74) Representative: Ling, Christopher John
IBM United Kingdom Limited,
Intellectual Property Department,
Hursley Park
Winchester, Hampshire SO21 2JN (GB)

(54) Dymamic translucent windows in a graphical user interface

(57) A method of and system (10) for implementing dynamic translucent windows (49) in a graphical user interface. Each translucent window (49) has associated therewith a foreground buffer (57) and a background buffer (59) . Whenever a translucent window is updated, the system (10), starting with the lowest z-order updated translucent window, combines (95)the updated translucent window's foreground (57) and background (59) buffers into a translucent image. If the translucent image is in a clip region (43), the system (10) displays (97) the portion of the translucent image in the clip region (43) and turns off (119) the update marker. If the translucent image is in a deferred clip region (55), the system copies (111) the portion of the translucent image in any deferred clip region (55) into the background buffer (59) or buffers of the translucent window (49) or windows that define the deferred clip region (55) and marks (113) the translucent window updated. The system then goes (117) to the next lowest z-order updated translucent window and repeats the process until the topmost window is reached, thereby to propagate the translucency effect up through the stack of windows.

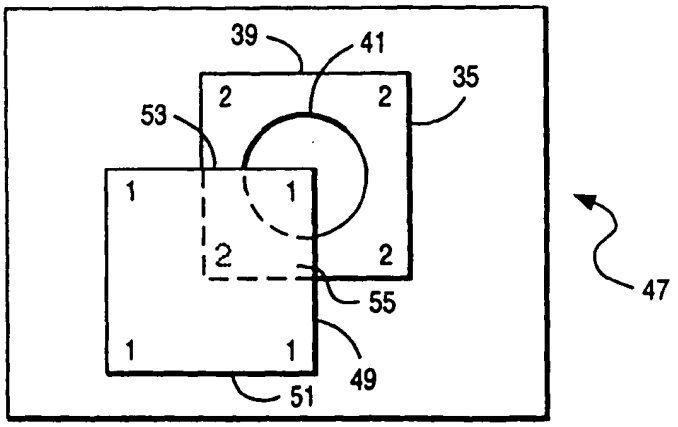


FIG. 3

EP 0 806 756 A2

Description

Field of the Invention

The present invention relates generally to computer system user interfaces and more particularly to an improved method of and system for implementing dynamic translucent windows in a computer system graphical user interface.

Background of the Invention

Graphical user interfaces in windowing environments can display multiple windows. Typically, each window displayed provides a user interface to a separate application. When several windows are displayed, they are usually displayed in overlapping fashion.

Graphical user interface systems typically provide only opaque windows. When one window overlays another window, the top window completely obscures the view of any portion of any windows that lie directly beneath it. The order in which windows are stacked on the display is referred to as "z-order." A window overlays any windows that are below it in z-order and is overlaid by any window that is above it in z-order.

The window at the top of the z-order is not obscured by any other windows and is usually the "active" window, in that it can receive user input. However, in multitasking environments, the windows beneath the top window in z-order are associated with active applications and they can be drawn into.

In systems with only opaque windows, the system does not allow material to be drawn into any portion of a window that is overlaid by a window above it in z-order. A window that is only partially obscured has a portion defined as its clip region that is visible on the display. Anything that is drawn into the clip region of a partially obscured window is displayed, but anything drawn outside the clip region is simply discarded.

A user may wish to see what is drawn into windows that are beneath other windows in z-order. A user may be alerted that something has been drawn into an obscured window by observing a change in the contents of the clip region of the obscured window, but the user must surface the obscured window in order to see its contents.

It is therefore an object of the present invention to provide translucent windows that allow a user to see material draw into windows that are covered by other windows. It is a further object of the present invention to provide translucent windows that are dynamically updated as material is drawn into windows that are covered by other windows.

Summary of the Invention

In accordance with the present invention, a translucent window is a window that shows those portions of any windows that lie beneath it, while still displaying the translucent window. The images of windows beneath the translucent window are combined with the image of the translucent window so that elements of both are visible at the same time. An opaque window is a window that completely obscures any portion of any window that lies beneath it.

Each translucent window has associated therewith a foreground buffer and a background buffer. The foreground buffer contains the native contents of the window, i.e. anything that is drawn into the window. The background buffer contains an image of all the windows that lie beneath the translucent window. Translucency is accomplished by combining the contents of the background and foreground buffers to form a translucent image.

A clip region is a portion of a window that is not obscured by any window above it in z-order. When a window is on the top of a stack of windows, the entire window is in its clip region. In accordance with the present invention, a deferred clip region is a portion of a window that is directly overlaid by a translucent window. Thus, a translucent window that directly overlays another window defines a deferred clip region in the overlaid window, and the overlaying translucent window may be referred to as the deferred clip region's translucent window. In the context of the present invention, directly overlaid means there are no other windows, translucent or opaque, between the window and the portion of the translucent window that overlays it.

Whenever an application draws into a window, if the window is opaque and has a clip region, then the system displays the drawing in the clip region. If the opaque window has a deferred clip region, then the system draws the drawing into the background buffer of the any translucent window that defines the deferred clip region and marks the translucent window updated.

If the window is translucent, then the system draws the drawing into the foreground buffer of the window and marks the translucent window updated.

Whenever a translucent window is updated, the system, starting with the lowest z-order updated translucent window, combines the updated translucent window's foreground and background buffers into a translucent image. If any of the translucent image is in the translucent window's clip region, the system displays the portion of the translucent image in the clip region and turns off the update marker. If any of the translucent image is in a deferred clip region, the system copies the portion of the translucent image in any deferred clip region into the background buffer or buffers of the translucent window or windows that define the deferred clip region and marks the translucent window updated. The system then goes to the next lowest z-order updated translucent window and repeats the process until the topmost window is reached, thereby to propagate the translucency effect up through the stack of windows.

Brief Description of the Drawings

The invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is block diagram of a computer system according to the present invention;

Figure 2 is a pictorial view of a computer display according to the prior art;

Figure 3 is a pictorial view of a computer display according to the present invention;

Figure 4 is a graphical representation the contents of a foreground buffer according to the present invention;

Figure 5 is a graphical representation the contents of a background buffer according to the present invention;

Figures 6A-C are a flowchart of a preferred software implementation of the present invention; and

Figure 7 is a flowchart of a preferred software method of updating the background buffers of translucent windows overlaying another translucent window according to the present invention.

Detailed Description of the Preferred Embodiment

Referring now to the drawings, and first to Figure 1, a block diagram of a personal computer with which the system and method of the present invention may be implemented is designated generally by the numeral 10. System 10 includes a processor 11, which includes a central processing unit (CPU) 15 and random access memory 13. System 10 also includes additional memory in the form of a disk storage device 17 and a floppy disk device 19. Floppy disk device 19 is adapted to receive a diskette 21 that has recorded thereon software including a software implementation of the present invention. System 10 also includes user interface hardware including a display 23, a keyboard 25, and a mouse 27. The system also includes a printer 29.

Referring now to Figure 2, there is shown a pictorial representation of a computer display screen, designated generally by the numeral 31, according to the prior art. Screen 31 has displayed thereon a first window 33 that partially overlays a second window 35. For purposes of illustration, first window 33 includes a window border 37 and it has drawn in each of its corners a numeral "1". Second window 35 includes a window border 39 and it has drawn in each of its corners a numeral "2" and in its centre a circle 41. First window 33 is opaque in the sense that it completely obscures the portion of second window 35 that it overlays.

The portion of second window 35 that is visible in Figure 2 is its clip region 43. Whenever the system draws into second window 35, anything in clip region 43 is displayed and anything outside clip region 43 is discarded. Since first window 33 is not obscured by any other window, its entire area comprises a clip region 45.

Referring now to Figure 3, there is shown a pictorial representation of a display screen, designated generally by the numeral 47, similar to screen 31 of Figure 2, except that it includes at least one translucent window 49 according to the present invention. Translucent window 49 is similar to first window 33 of Figure 2 in that it includes a window border 51 and it has a numeral "1" displayed at each of it corners. However, translucent window 49 also has displayed therein a phantom or translucent image 53 that shows the portion of second window 35 that translucent window 49 overlays.

According to the present invention, second window 35 includes, in addition to clip region 43, a deferred clip region 55. Deferred clip region 55 is defined by the portion of second window 35 that is overlaid by translucent window 49. For clarity of illustration only two windows are illustrated in the drawings, but those skilled in the art will recognize that in typical computer displays several windows may be stacked in overlapping fashion. In the general case, a deferred clip region is defined by that portion of a translucent window that directly overlays a window in the sense that there are no intervening windows in z-order between the deferred clip region and the translucent window that defines the deferred clip region. Thus, a single window may have multiple deferred clip regions defined by the portion of each translucent window that directly overlays it, but a deferred clip region of a window is defined by only a single translucent window.

Translucent windows according to the present invention include foreground buffers and background buffers. Referring to Figures 4 and 5, the contents of the foreground buffer for translucent window 49 of Figure 3 is represented in Figure 4 by the numeral 57, and the contents of the background buffer for translucent window 49 is represented in Figure 5 by the numeral 59. Foreground buffer 57 contains the native contents of translucent window 49. Foreground buffer 57 thus contains everything that the application associated with translucent window 49 draws into it.

Background buffer 59 contains an image of the windows that translucent window 49 overlays. Thus, background buffer 59 contains the corner of second window 35, including a portion 61 of window border 39, a numeral "2", and a portion 63 of circle 41. As will be discussed in greater detail below, background buffer 59 contains an image of the material drawn into the deferred clip regions defined by translucent window 49. Translucent window 49 is formed by combining buffers 57 and 59, using known colour combination techniques, to form a translucent image that is displayed on screen

47.

Referring now to Figures 6A-C, there is shown a flow chart of the logic of a preferred software implementation of the algorithm of the present invention that executes when a window finishes drawing. The process starts at block 63 and the system waits until something is drawn into a window. In the preferred embodiment the system waits at decision block 65 until a drawing finishes drawing. GUI systems typically have mechanisms for determining that a window is finished drawing. For example, the preferred OS/2 operating system issues the message WM_ENDPAINT when a window finishes drawing. The check for drawing ended is added for performance reasons, so as not perform the algorithm for every atomic drawing operation.

After a window finishes drawing, the system tests at decision block 67 whether anything has been drawn into the window's clip region. If so, the system tests at decision block 69 whether or not the window is opaque. If so, the system displays the drawing that is in the clip region immediately at process block 71. If at decision block 69 the window is not opaque, i.e. the window is translucent, the system adds the drawing that is in the window's clip region to the windows foreground buffer at process block 73 and marks the window as updated at block 75. Opacity or translucency is an attribute of a window that may be fixed when the window is created or alterable by a user.

Referring now to Figure 6B, the system tests at decision block 77 whether the window has any deferred clip regions, i.e. if it is overlaid by any translucent windows. If so, the system sets the current_region equal to the window's first clip region at block 79. The system then tests at decision block 81 whether anything is drawn in the current region. If so, the system puts that drawing into the background buffer of the current_region's translucent window at process block 83 and marks the current_region's translucent window as updated at block 85. The system then tests at decision block 87 whether the window has any more deferred clip regions. If so, the system sets the current_region equal to the window's next deferred clip region at process block 89 and loops back to decision block 81 to repeat the process until the system finds, at decision block 87, that the window has no more deferred clip regions.

Referring now to Figure 6C, after the system has handled all the deferred clip regions, the system sets trans_window equal to the translucent window lowest in z-order at process block 91. The system then tests at decision block 93 whether trans_window has been marked as updated. If so, the system combines the translucent_window's background and foreground buffers at process block 95 to form translucent_image and then, at process block 97, displays any portion of translucent_image that is in trans_window's clip region. Next, as indicated generally at process block 99, the system updates the background buffers of any translucent windows that overlay trans_window, the details of which are shown if Figure 7.

Referring to Figure 7, the preferred background buffer updating algorithm starts at block 101. The system tests at decision block 103 whether trans-window has any deferred clip regions. If not, the process exits at block 105 and returns to Figure 6B. If trans_window has any deferred clip regions, the system sets current_region equal to trans-window's first deferred clip region at process block 107. Then, the system tests at decision block 109 whether any portion of translucent_image is in current_region. If so, the system copies the portion of translucent_image that is in current_region into the background buffer of current_region's translucent window at process block 111 and marks current_region's translucent window as updated at block 113. The system then tests at decision block 115 whether trans_window has any more deferred clip regions. If so, the system sets current_region equal to trans_window's next clip region at block 117 and loops back to decision block 109 to repeat the process until the system finds, at decision block 115, that trans_window has no more deferred clip regions, at which time the system exits at block 105 and returns to Figure 6C.

Referring again to Figure 6C, after the system has completed the process step shown generally at block 99, the system turns off the update marker on trans_window at block 119 and tests at decision block 121 whether there are any more translucent windows above trans_window in z-order. If so, the system sets trans_window equal to the translucent window next highest in z-order at process block 123 and loops back to decision block 93 to repeat processing until all translucent windows in the stack have been handled, at which time the system returns to Figure 6A to wait for a window to finish another drawing.

Claims

1. A method of implementing translucent windows (49) in a computer system having a display (23), which comprises the steps of:

   for each window (35,49) displayed on said display, defining a clip region (43) that includes a portion of the window not overlaid by any other window and a deferred clip region (55) that includes a portion of the window directly overlaid by a translucent window (49);

   for each translucent window displayed on said display, defining a foreground buffer (57) that contains the contents of the translucent window and a background buffer (59) that contains an image of the deferred clip region of any window directly overlaid by the translucent window.

2. A method of implementing translucent windows as claimed in claim 1, further comprising the steps of:

whenever an application draws a drawing into a clip region (43) of a translucent window (49), adding (73) the drawing to the foreground buffer (57) of said translucent window;

whenever an application draws a drawing into a deferred clip region (55) of a window, adding (83) the drawing to the background buffer (59) of the translucent window that defines said deferred clip region and marking (85) the translucent window that defines said deferred clip region updated.

3. A method as claimed in claim 2, further comprising the step of:

combining (95) the background buffer (59) with the foreground buffer (57) of any translucent window (49) having a buffer that has been drawn into to create a translucent image, displaying (97) any portion of said translucent image that is within a clip region of said updated translucent window and copying (99) any portion of said translucent image that is within a deferred clip region of said updated translucent window into the background buffer of the translucent window that defines said deferred clip region.

4. A method as claimed in claim 1, further comprising the steps of:

whenever an application draws a drawing into a clip region of an opaque window, displaying (71) said drawing in said clip region;

whenever an application draws a drawing into a clip region of a translucent window (49), adding (73) the drawing to the foreground buffer (57) of the translucent window and marking (75) the translucent window updated;

whenever an application draws a drawing into a deferred clip region of a window, adding (83) the drawing to the background buffer (59) of the translucent window that defines said deferred clip region and marking (85) the translucent window that defines said deferred clip region updated;

for each updated translucent window, combining (95) the background buffer with the foreground buffer to create a translucent image, displaying (97) any portion of said translucent image that is within a clip region of said updated translucent window and copying (99) any portion of said translucent image that is within a deferred clip region of said updated translucent window into the background buffer of translucent window that defines said deferred clip region and marking (119) said translucent window as updated.

5. A method as claimed in claim 4 wherein:

said display (23) is adapted to display a plurality of windows arranged in a z order: and

said combining step is carried out on the translucent window (49) having the lowest z order in said display.

6. A method as claimed in claim 5, further comprising steps of:

unmarking (119) said updated lowest z-order translucent window; and,

repeating said combining step (95).

7. A system for implementing translucent windows (49) in a computer system (10) having a display (23), which comprises:

means for defining, for each window (35, 49) displayed on said display, a clip region (43) that includes a portion of the window not overlaid by any other window and a deferred clip region (55) that includes a portion of the window directly overlaid by a translucent window (49);

a foreground buffer (57) for each translucent window displayed on said display, said foreground buffer containing the contents of the translucent window; and

a background buffer (59) for each translucent window displayed on said display, said background buffer containing an image of the deferred clip region of any window directly overlaid by the translucent window.

8. A system as claimed in claim 7, further comprising:

means for adding (73), whenever an application draws a drawing into a clip region (43) of a translucent window (49), the drawing to the foreground buffer (57) of said translucent window;

means for adding (83), whenever an application draws a drawing into a deferred clip region (55) of a window, the drawing to the background buffer (59) of the translucent window that defines said deferred clip region and marking (85)

the translucent window that defines said deferred clip region updated.

9. A system as claimed in claim 8, further comprising:

means for combining (95) the background buffer (59) with the foreground buffer (57) of any translucent window (49) having a buffer that has been drawn into to create a translucent image;

means for displaying (97) any portion of said translucent image that is within a clip region of said updated translucent window; and,

means for copying (99) any portion of said translucent image that is within a deferred clip region of said updated translucent window into the background buffer of the translucent window that defines said deferred clip region.

10. A system as claimed in claim 7, further comprising:

means for displaying (71), whenever an application draws a drawing into a clip region of an opaque window, said drawing in said clip region;

means for adding (73), whenever an application draws a drawing into a clip region of a translucent window (49), the drawing to the foreground buffer (57) of the translucent window and marking (75) the translucent window updated;

means for adding (83), whenever an application draws a drawing into a deferred clip region of a window, the drawing to the background buffer (59) of the translucent window that defines said deferred clip region and marking (85) the translucent window that defines said deferred clip region updated;

means for combining (95), for each updated translucent window, the background buffer with the foreground buffer to create a translucent image;

means for displaying (97) any portion of said translucent image that is within a clip region of said updated translucent window;

means for copying (99) any portion of said translucent image that is within a deferred clip region of said updated translucent window into the background buffer of the translucent window that defines said deferred clip region; and

means for marking (119) said translucent window as updated.

11. A system as claimed in claim 10, wherein:

said display (23) is adapted to display a plurality of windows arranged in a z-order; and

said combining step (95) is carried on the translucent window (49) having the lowest z-order in said display.

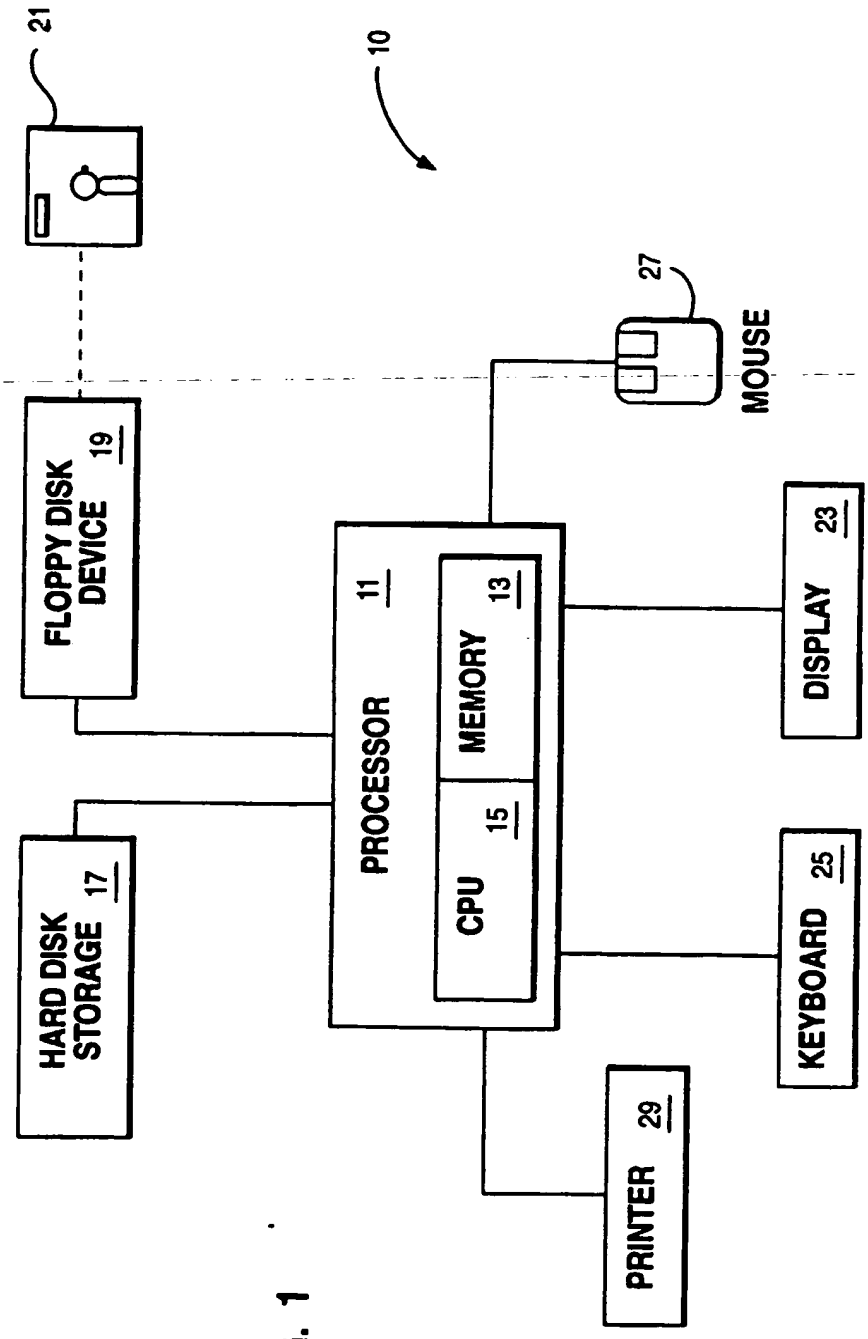

21
10
27
MOUSE
FLOPPY DISK DEVICE 19
PROCESSOR 11
MEMORY 13
CPU 15
DISPLAY 23
HARD DISK STORAGE 17
KEYBOARD 25
PRINTER 29

FIG. 1

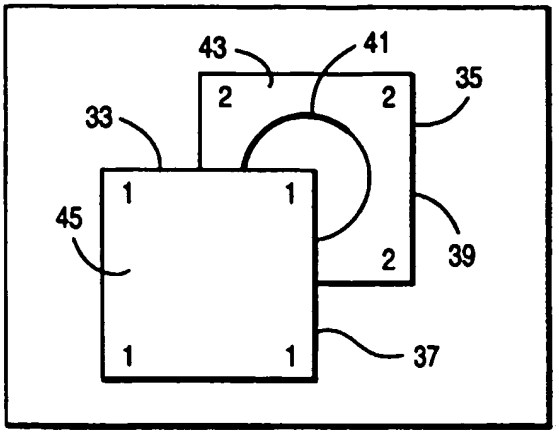


FIG. 2
PRIOR ART

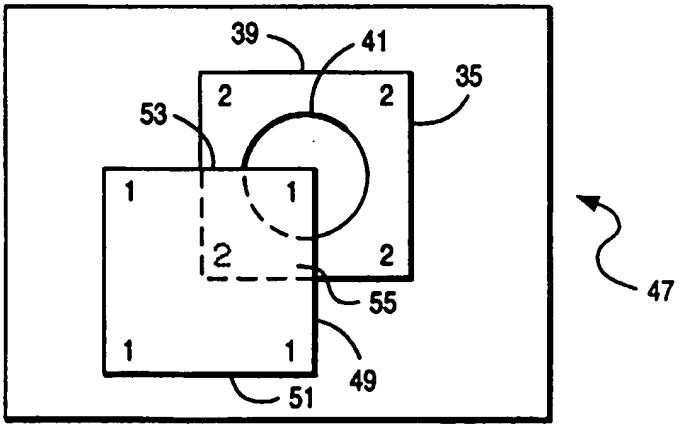


FIG. 3

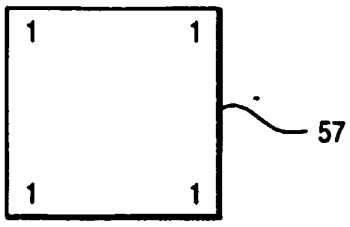


FIG. 4

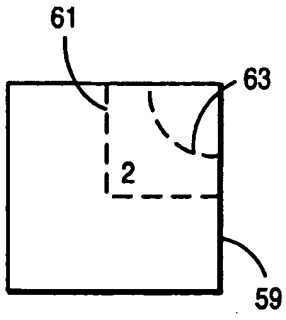


FIG. 5

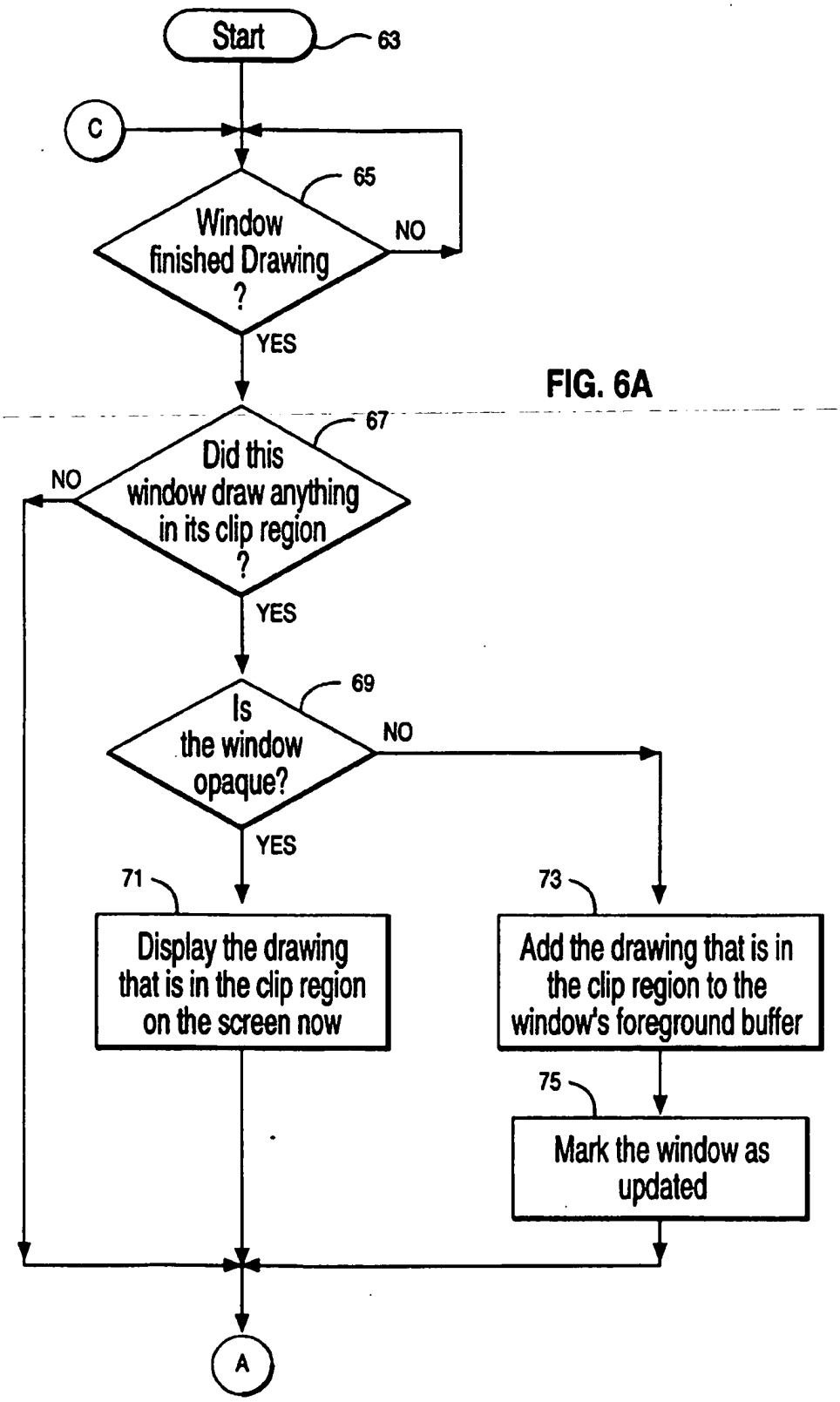
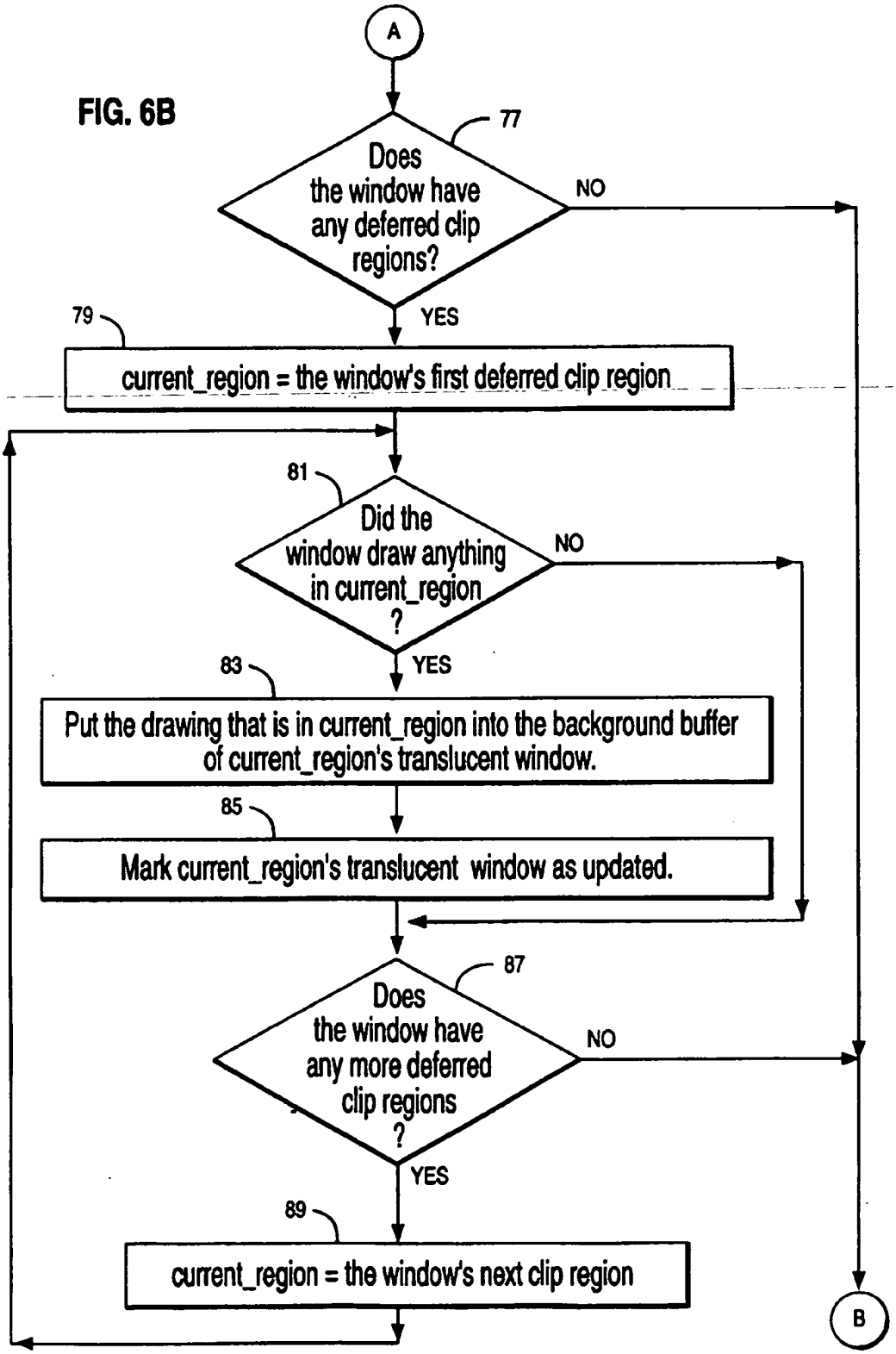

Start
63
C
65
Window finished Drawing ?
NO
YES
67
Did this window draw anything in its clip region ?
NO
YES
69
Is the window opaque?
NO
YES
71
Display the drawing that is in the clip region on the screen now
73
Add the drawing that is in the clip region to the window's foreground buffer
75
Mark the window as updated
A

FIG. 6A

FIG. 6B


A
77
Does the window have any deferred clip regions?
NO
YES
79
current_region = the window's first deferred clip region
81
Did the window draw anything in current_region ?
NO
YES
83
Put the drawing that is in current_region into the background buffer of current_region's translucent window.
85
Mark current_region's translucent window as updated.
87
Does the window have any more deferred clip regions ?
NO
YES
89
current_region = the window's next clip region
B

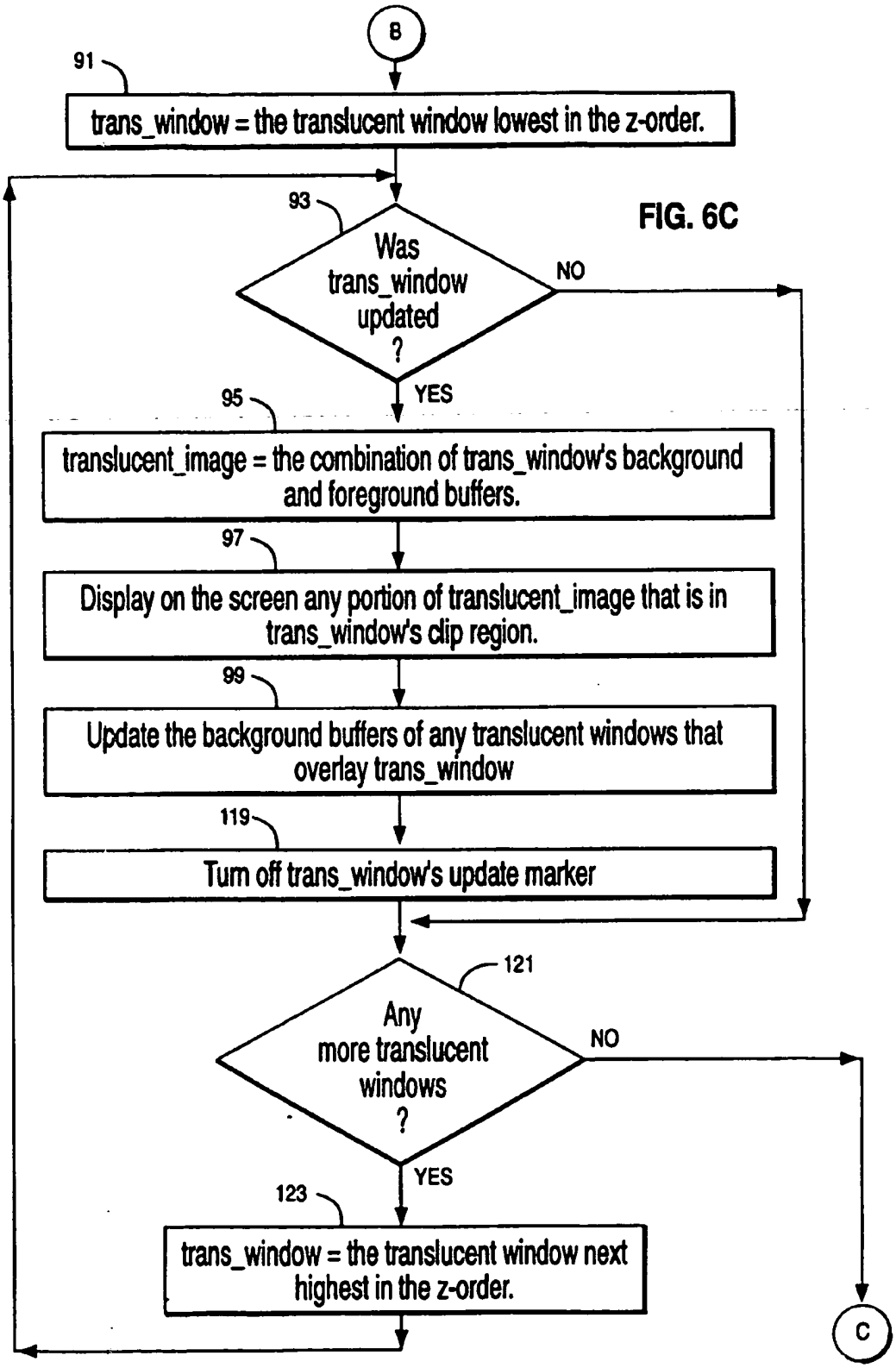
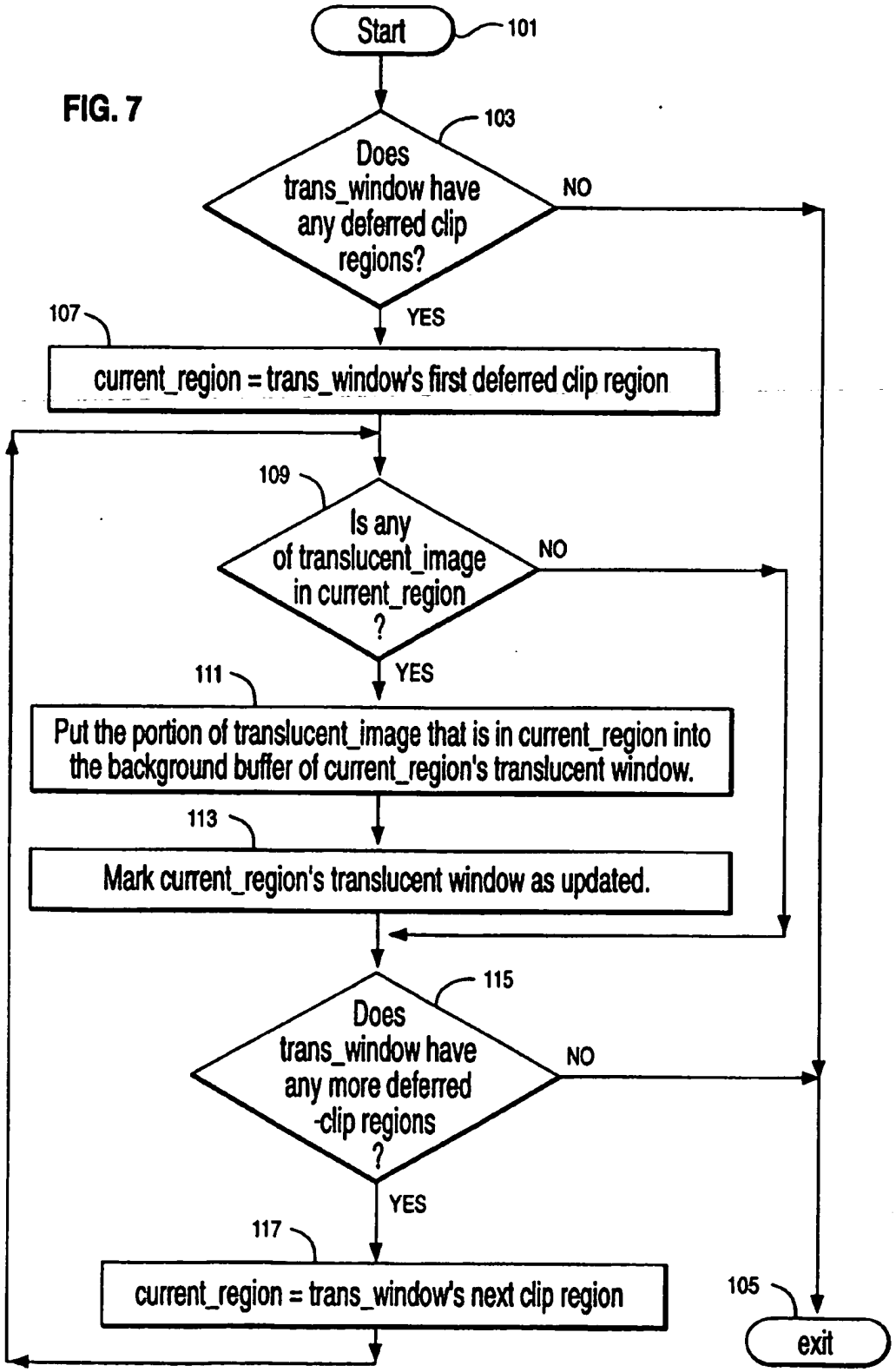

B
91
trans_window = the translucent window lowest in the z-order.
93
Was trans_window updated ?
NO
YES
95
translucent_image = the combination of trans_window's background and foreground buffers.
97
Display on the screen any portion of translucent_image that is in trans_window's clip region.
99
Update the background buffers of any translucent windows that overlay trans_window
119
Turn off trans_window's update marker
121
Any more translucent windows ?
NO
YES
123
trans_window = the translucent window next highest in the z-order.
C

FIG. 6C

FIG. 7


Start
101
Does trans_window have any deferred clip regions?
103
NO
YES
107
current_region = trans_window's first deferred clip region
109
Is any of translucent_image in current_region ?
NO
YES
111
Put the portion of translucent_image that is in current_region into the background buffer of current_region's translucent window.
113
Mark current_region's translucent window as updated.
Does trans_window have any more deferred -clip regions ?
115
NO
YES
117
current_region = trans_window's next clip region
105
exit